gitlab.local.dev

Author: Adrian DC

Description: Install, configure and maintain the gitlab.local.dev infrastructure with mise tasks

Date: 23/11/2026

Version: `2.0.0-43-g11c8590`

▸ 2. GitLab - Settings

Settings file

- **Configure the settings interactively, prefilled with the current values or defaults :**

```
gitlab.local.dev settings:configure
```

- **Show the current settings :**

```
gitlab.local.dev settings:show
```

- **Settings file location :** `~/.config/gitlab.local.dev/settings.env` (private, `600` permissions)
- Override the location with the `GITLAB_LOCAL_DEV_SETTINGS` environment variable
- Override any setting temporarily with its environment variable, for example `VERSION_GITLAB=19.6.1`

Task sources: `settings:configure`

```
#!/usr/bin/env bash

# Mise
#MISE alias="configure"
#MISE description="Configure the infrastructure settings interactively (prefilled with current values)"

# Configurations
set -e -o pipefail

# Sources
# shellcheck source=lib/common.sh
source "${MISE_PROJECT_ROOT:?}/lib/common.sh"

# Load current settings
settings_load
root_previous="${DOMAIN_ROOT}"

# Header
step "Configure settings in '${SETTINGS_FILE}'"
info 'GitLab releases: https://about.gitlab.com/releases/categories/releases/'
info 'GitLab upgrade path: https://gitlab-com.gitlab.io/support/toolbox/upgrade-path/'
info 'GitLab images: https://hub.docker.com/r/gitlab/gitlab-ce/tags'
info 'GitLab Runner releases: https://gitlab.com/gitlab-org/gitlab-runner/-/releases'
echo '' >&2

# Configure settings
for name in "${SETTINGS_NAMES[@]}"; do
  value="${!name}"

  # Follow the root domain changes for derived settings
  if [ "${name}" != 'DOMAIN_ROOT' ] && [ "${DOMAIN_ROOT}" != "${root_previous}" ] \
      && [ "${value}" = "$(DOMAIN_ROOT="${root_previous}" settings_default "${name}")" ]; then
    value=$(settings_default "${name}")
  fi

  # Show running versions
  case "${name}" in
    VERSION_GITLAB)
      if running=$(docker inspect --format '{{ .Config.Image }}' gitlab 2>/dev/null); then
        info "Running GitLab container image: ${running}"
      fi
      ;;
    VERSION_RUNNER)
      if running=$(gitlab-runner --version 2>/dev/null | sed -n 's/^Version: *//p') && [ -n "${running}" ]; then
        info "Installed GitLab Runner version: ${running}"
      fi
      ;;
    GITLAB_PATH)
      if running=$(gitlab_path_detect) && [ -n "${running}" ]; then
        info "Running GitLab container folder: ${running}"
      fi
      ;;
  esac

  # Ask setting value
  while true; do
    value=$(input "$(settings_description "${name}") [${name}]" "${value}")
    if settings_validate "${name}" "${value}"; then
      break
    fi
    warn "Invalid '${name}' value: '${value}'"
    if ! interactive; then
      exit 1
    fi
```

```
  done
  printf -v "${name}" '%s' "${value}"
done

# Write settings
mkdir -p "$(dirname "${SETTINGS_FILE}")"
temporary=$(mktemp "${SETTINGS_FILE}.XXXXXX")
{
  echo '# gitlab.local.dev settings (generated by settings:configure)'
  for name in "${SETTINGS_NAMES[@]}"; do
    printf 'export %s=%q\n' "${name}" "${!name}"
  done
} >"${temporary}"
chmod 600 "${temporary}"
mv -f "${temporary}" "${SETTINGS_FILE}"

# Result
echo '' >&2
info "Settings saved in '${SETTINGS_FILE}'"
"${MISE_PROJECT_ROOT}/.mise/tasks/settings/show"
```

Table of contents

- Settings file
- Available settings
- GitLab versions
- Recommended configurations

Available settings

Variable	Value	Defaults or examples
DOMAIN_IP	The GitLab remote IP address	`10.16.0.20`
DOMAIN_ROOT	The remote root domain name	`local.dev`
DOMAIN_GITLAB	The GitLab remote domain name	`gitlab.${DOMAIN_ROOT}`
DOMAIN_PAGES	The GitLab Pages remote domain name	`pages.${DOMAIN_ROOT}`
DOMAIN_REGISTRY	The GitLab Container Registry remote domain name	`registry.${DOMAIN_ROOT}`
DOMAIN_CONTACT	The domain public contact email	`contact@${DOMAIN_ROOT}`
DOMAIN_LOCATION	The domain public "C/ST/O/L" location	`C=FR/ST=France/O=GitLab/L=Paris`
PORT_GITLAB	The GitLab HTTPS remote port number	`8343` or `443`
PORT_PAGES	The GitLab Pages remote port number	`8390` or `8090`
PORT_REGISTRY	The GitLab Container Registry remote port number	`8350` or `5000`
PORT_SSH	The GitLab SSH remote port number	`8322` or `22`
ACCESS_MODE	The network mode (IP address, /etc/hosts or DNS)	`ip`, `hosts` or `dns`
RUNNER_CACHE_DISABLED	The GitLab Runner cache deactivation	`false` or `true`
RUNNER_CONCURRENT	The GitLab Runner maximum concurrent jobs number	`4` or `1`
SSL_LETSENCRYPT	The optional Let's Encrypt certificates activation	`false` or `true`

VERSION_GITLAB	The GitLab release version to run	`19.6.1`
VERSION_RUNNER	The GitLab Runner release version to run	`19.6.0`
VERSION_VIRTUALBOX	The VirtualBox major release version (host machines)	`7.0`
BACKUP_GITLAB_UPDATE	The GitLab data backup upon server updates	`true` or `false`
GITLAB_PATH	The GitLab server folder (empty to detect from container)	`/home/administrator/gitlab`

- **One-time server certificates flags :**
- `gitlab.local.dev server:certificates --refresh-ssl` : Regenerate SSL certificates (or `REFRESH_SSL=true`)
- `gitlab.local.dev server:certificates --refresh-dns` : Reconfigure DNS authentication (or `REFRESH_DNS=true`)

GitLab versions

- **Check GitLab releases :** GitLab Releases
- **Check GitLab upgrade path :** GitLab Upgrade Path tool
- **Check GitLab images :** DockerHub gitlab/gitlab-ce
- **Check GitLab Runner releases :** GitLab Runner releases

> ⚠ **Required upgrade stops**
>
> GitLab upgrades must go through each required upgrade stop (for example `18.5`, `18.8`, `18.11`, `19.2`, `19.5`), with `gitlab.local.dev server:update` for each stop, and background migrations finished in `Admin Area / Monitoring / Background migrations` before the next stop.

Recommended configurations

- **Machines sizes :**
- **GitLab server :** `128 GB`
- **GitLab Runner :** `48 GB`
- **Host machine configurations :**
- **VirtualBox VMs location :** `/home/services/gitlab.local.dev`

- **Default network configurations :**
- **Host machine :** `10.16.0.1`
- **GitLab server :** `10.16.0.20`
- **GitLab Runner :** `10.16.0.21`
- **Ubuntu installation configurations :**
- **Encrypt the LVM group with LUKS :** Enable
- **Passphrase :** Specific to project
- **Confirm passphrase :** Specific to project

Copyright © 2026 - Adrian DC

gitlab.local.dev